Reprint

Multiple VMEbus Interconnections Using Hardware Semaphores

T. Antonakopoulos, N. Kanopoulos and V. Makios

ISMM International Symposium for Mini and Microcomputers and their Applications

LUGANO, SWITZERLAND, JUNE 1990

Multiple VMEbus Interconnections Using Hardware Semaphores.

T. Antonakopoulos *, N. Kanopoulos **, V. Makios *

* Laboratory of Electromagnetics, School of Engineering, University of Patras, Greece.

** Center for Digital Systems Research, Research Triangle Institute (RTI), NC, USA.

Abstract: **In many VMEbus based systems, the internal system bus is the most commonly shared resource of the system and can create a performance bottleneck. In this paper, various techniques of logical bus interconnections using hardware semaphores are discussed and the proposed interconnection mechanism is explained. The system's performance improvement is evaluated using the two bus architecture instead of the single bus configuration.**

I. INTRODUCTION

The need of using high processing power systems in many commercial and industrial applications as well as the decreasing cost of the new microprocessors stimulate the design and implementation of multiprocessor systems. Effective implementation of these systems can be achieved if the computational problem can be decomposed to profit from the parallelism of the system and the system overhead, which is due to the processors cooperation, is kept low [1], [2]. This overhead is mainly affected from the usage contention of a limited number of common resources as the system's bus. The system bus, which is a time-shared resource, is used for exchanging all the data and control information and interconnects the processing units, the I/O lines and the system memory [3]. Syncronization and control of the system's bus usage is achieved using simple and fast arbitration protocols, usually implemented by hardware [4]. These protocols ensure that low priority processors gain access to the bus without being completely locked out.

To resolve the bus contention problems, many computer bus architectures and arbitration protocols have been proposed both in theoretical and practical view. The most known is the loosely coupled architecture, where each processor is ignorant of the other's existence and communicates with them using a common memory. This architecture is the most prefered in VMEbus based systems where the VMEbus is used as the global passing bus [5].

In this paper, the VMEbus standard is described and a new technique is proposed to increase the system's performance, by reducing the bus contention. In Section II, the VMEbus architecture is briefly described and the need for new system considerations is highlighted. This Section also discusses various techniques for logical bus interconnections, while in Section III, the presented performance analysis evaluates the system's performance improvement.

II. BUS INTERCONNECTIONS.

The VMEbus is the fastest growing industry bus standard. It is designed as a global parallel interconnect, supporting different types of data transfers, using a non-multiplexed asynchronous protocol. It has four functional groups, the data transfer bus (DTB), the DTB arbitration, the interupt bus and the utility bus [5]. In every VMEbus based system there are four arbitration levels and in the same arbitration level, the processors follow a daisy-chain priority scheme. The VMEbus has seven interrupt levels and allows the use of distributed interrupt handling. In Fig. 1, a typical VMEbus based system architecture is shown. In each VMEbus, there is only one system controller and multiple MASTER and SLAVE board interfaces. A MASTER initiates and controls any data transaction across the bus, while a SLAVE responds to a data transaction on a passive way.

In order to increase the system's performance a secondary bus is used like VMX, VSB etc [5]. That allows

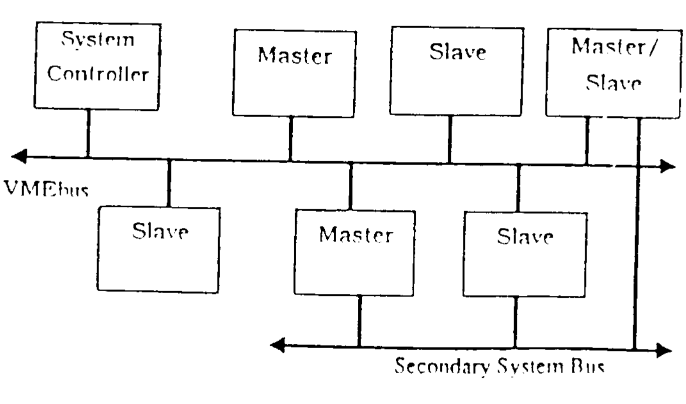

Fig. 1. VMEbus System Architecture.

parallel transfer accross the system, especially in systems where multiple processors work together executing cooperating tasks. During task execution, there are processing periods that do not require access across the VMEbus or the subsystem bus and there are transfer periods that require the bus control to pass the respective message.

When contention occures, the lower priority processor(s) are queued, waiting to gain the control of the bus. Contention can be minimized using two approaches: first, to increase the bus transfer speed and second, to use multiple buses. The bus transfer speed is limited by the processor's speed and its influence decreases as the number of processors increases. The use of multiple buses has the disadvantage of requiring multiple interfaces and the hardware overhead becomes unacceptable. In order to avoid this disadvantage, we propose a new system architecture where only one bus type is used and allows the dynamic use of multiple subsystem buses. In that architecture the processing modules of the system are organized in functional groups and each group has a dedicated VMEbus for intergroup communication. Each subsystem is independent of the others and its throughput is not affected by the others traffic.

When a processor from a group has to communicate with a module in another group, it connects the buses using the respective Intermediate Interconnection Module (IIM). The IIM has two modes of operation, the buffering and the connecting mode. At the buffering mode the buses are isolated and The IIM acts as the bus controller for the second bus. At the connecting mode the IIM's arbiter is not active, the two VMEbuses are connected and act as a single bus. After the completion of the communication, the interconnection is released allowing parallel data transfers inside these groups. Fig. 2 shows a triple bus system architecture. In (a) is shown the physical system architecture, while in (b) - (e), are shown the possible logical system architectures.

The interconnection function is achieved using a Read_Modify_Write (RMW) cycle to access the hardware semaphore which belongs to the respective IIM. During the Read part of the cycle, the processor reads the semaphore's value and at the write part, it sets it. If the semaphore was already set, the processor has to wait for a period of time prior to begin a new attempt, because the interconnection is under the control of another processor. If the semaphore was free, the processor takes the control of the interconnection at the end of the RMW cycle. The interconnection is released when the processor clears the semaphore to allow the establishment of a new interconnection. Fig. 3 shows the timing diagram of that operation in a logic analysis form. Processor (F) reads and sets the semaphore (SEMA) and after a certain time releases it. Processor (S) reads the semaphore, sets it but, because of its value, does not take the control and has to try again after a while. For systems with multiple interconnectable VMEbuses, the memory mapping has to follow certain rules to avoid memory conflicts during interconnection. The same holds for the interrupt structure, especially when a distributed interrupt structure is used.

The above described operation is for a single value semaphore which is controlled only from the processor. The disadvantage of this operation is that, while the two buses are connected, only one processor can use the two subsystems and all the other processors have to wait to gain the semaphore's control for intergroup transactions. The use of a multiple value semaphore can overcome this disadvantage. For this type, the semaphore does not take only the values "0" and "1", but it has an assosiated counter, which increases each time a processor requests bus connection and decreases each time a processor clears the respective connection. Using this type of semaphore the processor's queueing is minimized and the system performance increases.

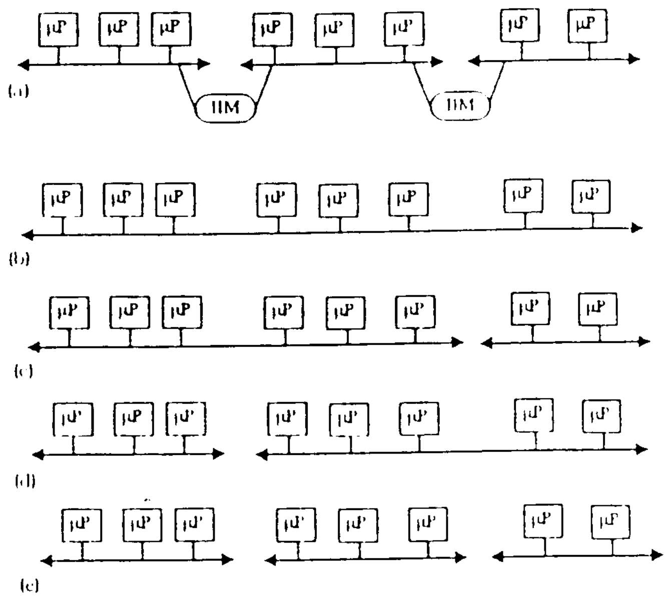

Fig. 2. Triple Bus System Modes.

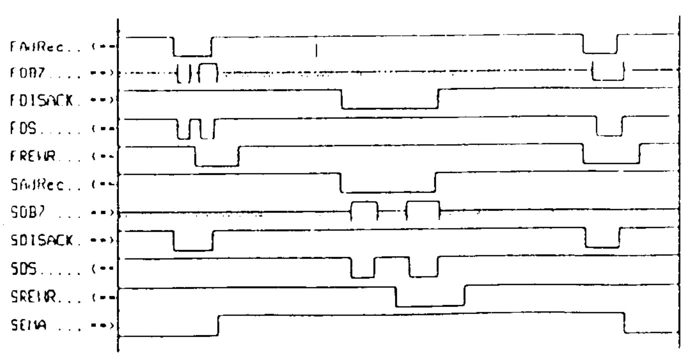

Fig. 3 The Single Value Semaphore Operation

III PERFORMANCE ANALYSIS.

In multiprocessor systems, the important performance measures are the data throughput and the processor access time [4]. In order to analyze the performance of a single VMEbus and to compare it with a two interconnected buses system, the following assumptions are made:

- The interarrival time of bus access requests for each processor is greater than the total service time and it is represented as an exponentially distributed random variable.
- The same number of processors is concerned with the same workload.
- All processors request access to the bus with equal probability.
- When the bus is available, the highest priority requesting processor accesses the bus immediately.
- If the bus is unavailable, the requesting processors become idle until the bus is free.
- The processors follow the "release when done" mode.

i) Single Bus Architecture [4].

We suppose that there are n processors in the bus with different priorities (either due to the different arbitration level or due to the daisy-chain scheme). The processors are identical and from the bus point of view, the probability that a processor requests service is equal to the traffic intensity (ρ) of that processor. The processor n has the highest priority, while processor 1 has the lowest.

The probability that processor i successfully accesses the bus is:

$$p_i = \sum_{k=0}^{i-1} \text{Prob[} k \text{ processors request access } | \text{ processor } i \text{ requests access]}$$

$$= \sum_{k=0}^{i-1} \text{Prob[} k \text{ processors request access]}$$

$$= (1-\rho)^{n-i} \sum_{k=0}^{i-1} [\binom{i-1}{k} \rho^{k} (1-\rho)^{i-k-1}]$$

$$= (1-\rho)^{n-i} \qquad (1)$$

If μ is the service rate of the bus, then the access time for processor i is given by:

$$t_i = \frac{1}{\mu} \cdot \sum_{j=1}^{\infty} \text{Prob[processor } i \text{ succeeds on the } j\text{th trial]}$$

$$= \frac{1}{\mu} \cdot \frac{1-p_i}{p_i} = \frac{1}{\mu} \cdot \frac{1-(1-\rho)^{n-i}}{(1-\rho)^{n-i}} \qquad (2)$$

and the normalized access time, T_i, is:

$$T_i = \mu . t_i = \frac{1-(1-\rho)^{n-i}}{(1-\rho)^{n-i}} \qquad (3)$$

The processor's throughput, s_i, is defined as the average number of succesful bus transactions per unit time, so:

$$s_i = \text{(processor message arrival rate) x (Prob. of success)}$$

$$= \lambda . \frac{1}{\mu . t_i} = \frac{\rho}{t_i} \qquad (4)$$

and the normalized throughput, S_i, which represents the fraction of time that the bus services the processor, is:

$$S_i = \frac{\rho}{T_i} = \frac{\rho (1-\rho)^{n-i}}{1-(1-\rho)^{n-i}} \qquad (5)$$

ii) Double Bus Architecture

As mentioned in Section II, the double bus architecture has two possible configurations, the single bus during interconnection and the double bus when the two buses are not connected. We define as q the probability that the system uses the double bus architecture and 1 - q as the probability that the architecture is single bus. In these two architectures, the priorities of the same processor are different, depending on his physical position and on his arbitration level. We define that the priority of processor i in the single bus architecture, becomes j in the double bus architecture where j is given by:

$$j = f[i, g(i,k)] \qquad (6)$$

where g(i,k) is the processors distribution function in the double bus architecture. Each processor in the system is determined by the (i,j,k) notation, which means that this processor has priority i in the single bus system and j priority in the k subsystem. In this case, k takes the values 0 or 1, for the first or the second bus respectively.

The probability that processor (i,j,k) successfully accesses the bus is given by:

$$P_{i,j,k} = (1 - q) \cdot p_i + q \cdot p_{j,k} \quad (7)$$

where the first term is the probability that the processor accesses the bus in the single bus mode, while the second term represents the probability that the processor access the respective subsystem in the double bus mode. From equation (1) we derive p_i while $p_{j,k}$ is given by an expression similar to (1), which is:

$$p_{j,k} = (1 - \varrho)^{\max[g(i,k)] - j} \quad (8)$$

and

$$P_{i,j,k} = (1 - q)(1 - \varrho)^{n - i} + q(1 - \varrho)^{\max[g(i,k)] - j}$$

$$= (1 - \varrho)^{n - i} + q \cdot \left[(1 - \varrho)^{\max[g(i,k)] - j} - (1 - \varrho)^{n - i} \right] \quad (9)$$

The processor's access ability is a comperative attribute which is improved when the number of processors in the system decreases. That means that the processor's priority distance from the highest priority decreases or at least remains the same, which in mathematical form is expressed by the following relation:

$$\max[g(i,k)] - j \leq n - i \quad (10)$$

From equations (9) and (10) is concluded that the access probability is increased using the interconnection mechanism. The achieved improvement depends on the subsystems interconnection probability and on the alteration of the priority distance. A new function determines, the system's reconfiguration impact function $R(i,j,k,\varrho)$, which is given by:

$$R(i,j,k,\varrho) = \frac{(1 - \varrho)^{\max[g(i,k)] - j}}{(1 - \varrho)^{n - i}} \quad (11)$$

and

$$P_{i,j,k} = p_i \cdot [1 + q \cdot (R(i,j,k,\varrho) - 1)] \quad (12)$$

The variation of the successfull access probability is given by:

$$\frac{\Delta P_i}{P_i} = \frac{P_{i,j,k} - p_i}{p_i} = q \cdot (R(i,j,k,\varrho) - 1) \quad (13)$$

Following the analysis above, the normalized access time for processor (i,j,k) becomes:

$$T_{i,j,k} = \frac{1 - p_{i,j,k}}{p_{i,j,k}} \quad (14)$$

while the normalized throughput is given by:

$$S_{i,j,k} = \frac{\varrho}{T_{i,j,k}} = \frac{\varrho P_{i,j,k}}{1 - p_{i,j,k}} \quad (15)$$

The system's performance improvement (SPI) is the ratio of the normalized throughput of the double bus architecture versus the normalized throughput of the single bus architecture, so:

$$SPI = \frac{S_{i,j,k}}{S_i} = \frac{p_{i,j,k} \cdot (1 - P_i)}{(1 - p_{i,j,k}) \cdot P_i} \quad (16)$$

SPI also depends on the intergroup traffic load, on the way the system has been reconfigured and on the processor's workload.

IV. CONCLUSIONS

In this paper, we have described a new interconnection technique for commercial VMEbus based systems to minimize the effect of the bus contention to the total system's performance. This technique simplifies the required hardware while the software overhead is kept low, simple and almost transparent to the application sofware. From the presented performance analysis, it is implied that the performance improvement depends strongly on the processors distribution and on the traffic load for intergroup communication.

REFERENCES

[1] M.A. Massan, G. Balbo and G. Conte: "Comparative Performance Analysis of Single Bus Multiprocessor Architectures", IEEE Trans. on Computers, Vol 31, No 12, December 1982, pp. 1179-1191.

[2] B.A. Bowen and R.J.A. Buhr: "The Logical Design of Multiple-Microprocessor Systems", Prentice-Hall Inc, New Jersy, 1980.

[3] M.H. Woodbury and K.G. Shin: "Performance Modeling and Measurement of Real Time Multiprocessors with Time-Shared Buses", IEEE Trans. on Computers, Vol 37, No2, February 1988.

[4] F.E. Guibaly: "Design and Analysis of Arbitration Protocols" IEEE Trans. on Computers, Vol 38, No2, February 1989, pp 161-171.

[5] S. Heath: "VMEbus User's Handbook", Heinemann Newner, Oxford, 1989.